



# Unidad de Control

Marisa Córdoba



## Unidad de Control

- Índice
  - ▶ Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la UC
  - UC microprogramada
  - Control de excepciones



## Objetivos

- Visión dinámica del computador:
  - mostrar cómo se ejecutan las instrucciones
  - describir el órgano encargado de que esto se lleve a cabo (la unidad de control)
  - entender esta ejecución de instrucciones mediante su representación en el tiempo por cronogramas
- Comprender el diseño de la unidad de control
  - Ser capaz de desglosar las instrucciones en operaciones básicas según la estructura del computador
  - Diseño de la unidad de control microprogramada
- Comprender y analizar aspectos de diseño que pueden mejorar el rendimiento a través de mejoras en la estructura del computador o de la UC

Unidad de Control

3



## Bibliografía

- de Miguel, P. *"Fundamentos de los computadores"*, Paraninfo, 2004. 9ª edición
- Patterson, D. A.; Hennessy, J. L. *Computer Organization and Design*. Morgan-Kaufmann. 2009. 4ª edición
- Stallings, W. *"Organización y arquitectura de computadores"*, Prentice Hall, 2006, 7ª Edición

Unidad de Control

4

## Introducción

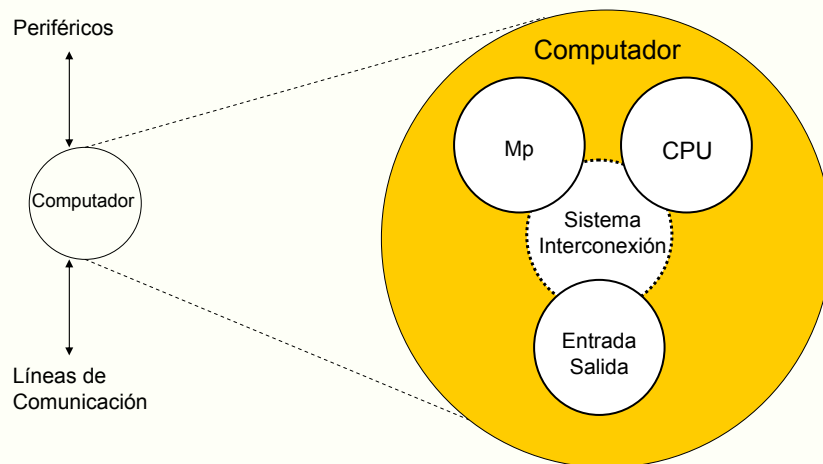
- Estudiaremos:
  - **Unidad de control:**
    - Encargada de interpretar las instrucciones del programa y gobernar la ejecución de las mismas
  - **Camino de datos:**
    - Parte de la CPU donde se transfieren los datos procedentes de la memoria o registros internos, para obtener los resultados
    - Debe soportar el conjunto de operaciones que precisan las instrucciones del repertorio capaz de interpretar la unidad de control
- Organización de procesadores: ha evolucionado
  - desarrollo tecnológico
  - la necesidad de obtener altas prestaciones

Unidad de Control

5

## Introducción

### Visión global del computador y de la CPU

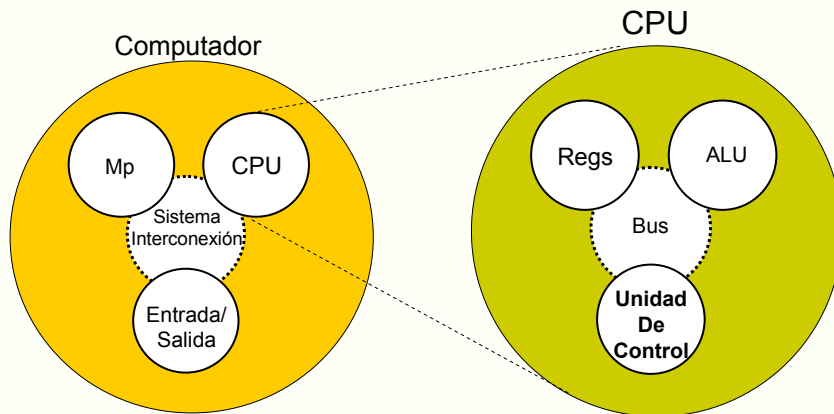


Unidad de Control

6

## Introducción

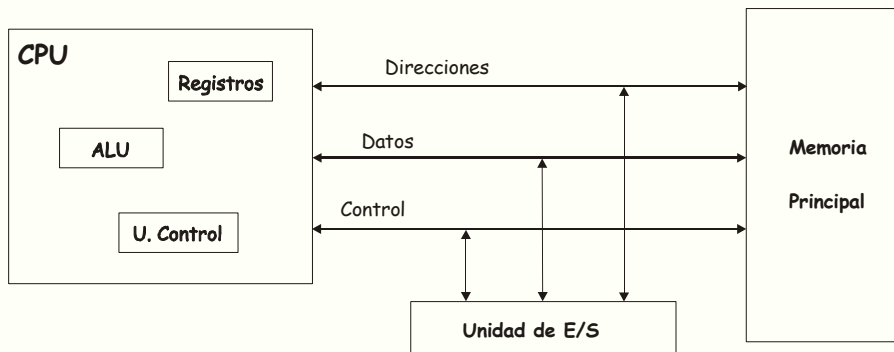
### Visión global del computador y de la CPU



Unidad de Control

7

## Esquema básico del computador Von Neumann. Componentes



Unidad de Control

8



## Introducción

- Función básica de CPU:  
**ejecutar instrucciones**
  1. Leer la instrucción en memoria
  2. Decodificar la instrucción  
para determinar qué acciones se han de realizar
  3. Ejecución: como máximo
    - Búsqueda de operandos
      - cálculo de operando
      - leer operando en memoria
    - Operación
    - Almacenar resultado
  4. Preparar la siguiente instrucción (ir a paso 1)

Unidad de Control

9



## Introducción

- Funciones de la CPU:
  1. Ejecuta instrucciones (función básica)
    - Lectura, decodificación, e interpretación de las instrucciones
    - Generación de órdenes para la ejecución
    - Secuenciamiento de las instrucciones  
(decidir cuál es la siguiente a ejecutar)
  2. Resuelve situaciones anómalas  
(desbordamiento, operación no válida, error de paridad, etc)
  3. Controla la comunicación con periféricos

Unidad de Control

10

## Introducción

- Entradas y salidas de la UC:
  - Entradas:
    - Registro de Instrucción: CO, MDs
    - Reloj: registro contador de fases
    - Registro de estado
    - Señales de control externas (E/S, Mem, )
  - Salidas
    - Todas las señales de control que permiten realizar cada una de las instrucciones máquina

## Introducción

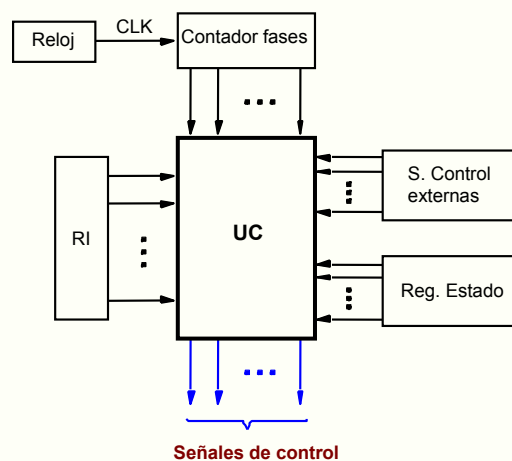
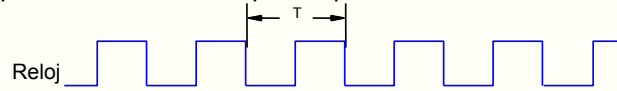


Figura Entradas y salidas de la UC.

## Introducción

- **Reloj:** tren de pulsos caracterizado por su periodo



- **señales de control:** siempre sincronizadas con el reloj
  - Define el tiempo de cada operación.
    - Memoria: tiempo de lectura y escritura
    - ALU: tiempo de operación
  - **CAMINO CRÍTICO:** camino de máximo retardo entre un origen y un destino. Depende de los dispositivos que tengan que atravesar las señales.
- Ej: si  $f=100$  MHz  
 $T=10$  ns y conviene que una operación aritmética se realice en un tiempo algo menor que 10 ns  
 Y Mem?

## Introducción

### Ciclo de lectura

Se supone AR siempre disponible en el bus de direcciones de memoria

1. dirección  $\rightarrow$  AR
2.  $M(AR) \rightarrow$  Reg; lectura

lectura: MEMRQ y RD  
 Reg o (DR)

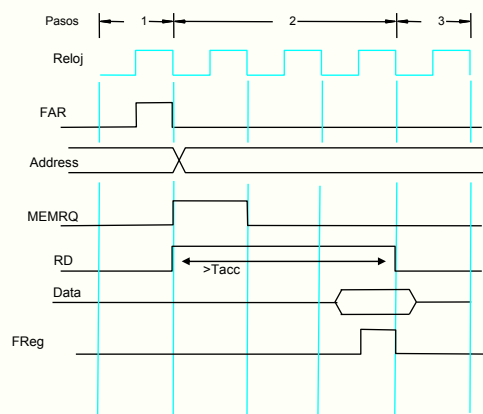


Figura . Temporización de la lectura en memoria

## Introducción

### Ciclo de escritura

AR y DR disponibles  
en buses de dir y dat

1. dirección → AR
2. dato → DR
3. DR → M(AR) ; escritura

escritura: MEMRQ y WR

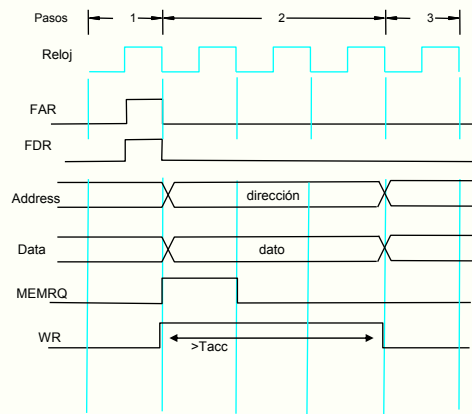


Figura . Temporización de la escritura en memoria

Unidad de Control

15

## Introducción

### Ciclos de lectura y escritura: **ciclo de BUS**

- implica un acceso al exterior a la CPU
  - durante ese tiempo sólo la CPU puede acceder a los buses
  - (ningún periférico)
- suele durar más de 1 ciclo de reloj (determinado por el camino crítico) interno a la CPU

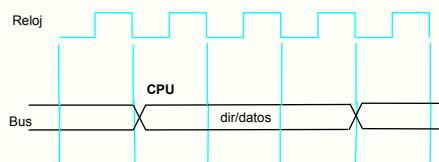


Figura . Ciclo de bus

Unidad de Control

16





## Introducción

### Evaluación del rendimiento

- Supóngase los tiempos de ejecución:
  - Acceso a memoria: 8 ns
  - ALU y sumadores: 2 ns
  - Acceso a registros: 1 ns
- ¿Cuál de las siguientes realizaciones será más rápida?
  - Una en la que cada instrucción se ejecuta en un ciclo de tamaño fijo (cada instrucción tarda lo que tardaría la más lenta).
  - Una realización donde cada instrucción se ejecuta en un ciclo de longitud variable (cada instrucción tarda únicamente lo necesario)
- ¿De qué dependería la duración del periodo de reloj?

Unidad de Control

17



## Unidad de Control

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la UC
  - UC microprogramada
  - Control de excepciones

Unidad de Control

18



## Operaciones Elementales

- Ciclo de instrucción: conjunto de acciones que requiere la ejecución de una instrucción
  1. Leer la instrucción en memoria
  2. Decodificar la instrucción
  3. Ejecución: como máximo
    - Búsqueda de operandos, que puede conllevar
      - Cálculo de operando
      - Leer operando en memoria
    - Operación
    - Almacenar resultado
  4. Preparar la siguiente instrucción (ir a paso 1)



## Operaciones Elementales

- Las **fases de ejecución** ayudan a simplificar el diseño de la UC.
  - Ej: fetch común a todas las instrucciones
- Funcionamiento del computador durante la ejecución de un programa consiste en una **sucesión de ciclos de instrucción**



## Operaciones Elementales

- **OPERACIONES ELEMENTALES** (microoperaciones):  
operaciones (realizables directamente por el hardware)  
en que la UC divide cada una de las fases de ejecución  
de una instrucción
- Las operaciones elementales se realizan por la UC  
mediante la activación de señales de control
- Cada microoperación dura un ciclo de reloj  
(excepto la Mem)
- Se pueden simultanear en el tiempo, cuidando:
  - Orden preestablecido
  - Conflictos

Unidad de Control

21



## Operaciones Elementales

- Tipos:
  - De transferencia
  - De proceso

Unidad de Control

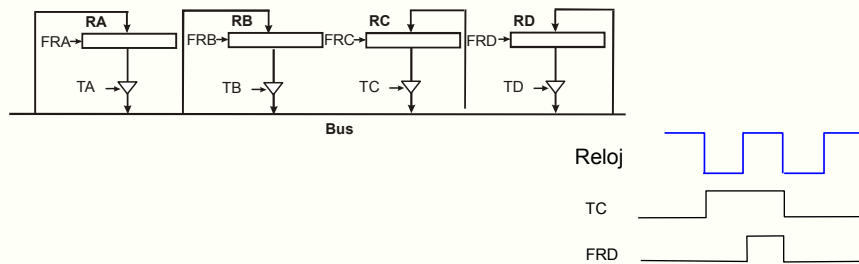
22



## Operaciones Elementales

### OE de transferencia

- Llevar información de un ORIGEN a un DESTINO
    - Establecer camino físico entre salida de origen y entrada de destino
    - Enviar señal al destino para que tome la información
- Ej: Transferencia a través de un bus de dos registros  $C \rightarrow D$



Unidad de Control

23



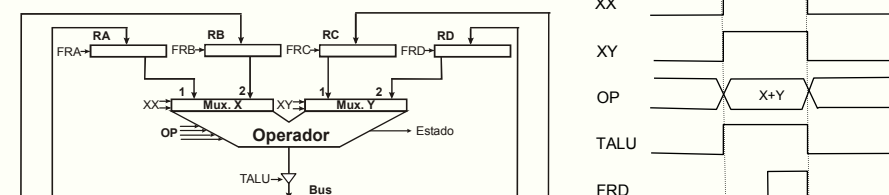
## Operaciones Elementales

### OE de proceso

- Funcionamiento similar a la transferencia, pero la información de ORIGEN se pasa por un operador que la procesa en su camino al DESTINO

Ej: Operación en la ALU con registros

$$A + C \rightarrow D$$



Unidad de Control

24



## Operaciones Elementales

### ■ Reglas de agrupación de $\mu$ operaciones

- Respetar orden preestablecido de algunas acciones  
Ej: antes de leer, guardar dirección en AR
- Evitar conflictos en el mismo recurso físico  
un dispositivo no puede estar en dos estados diferentes al mismo tiempo  
Ej: conflicto en bus (dos op. elem. intentan usar el mismo bus);  
memoria no puede leer y escribir al mismo tiempo
- La información debe guardarse en algún sitio: ORIGEN y DESTINO deben poder almacenar información (reg o mem)  
Ej: no se deben simultanear con el mismo destino si la información es distinta

Unidad de Control

25



## Unidad de Control

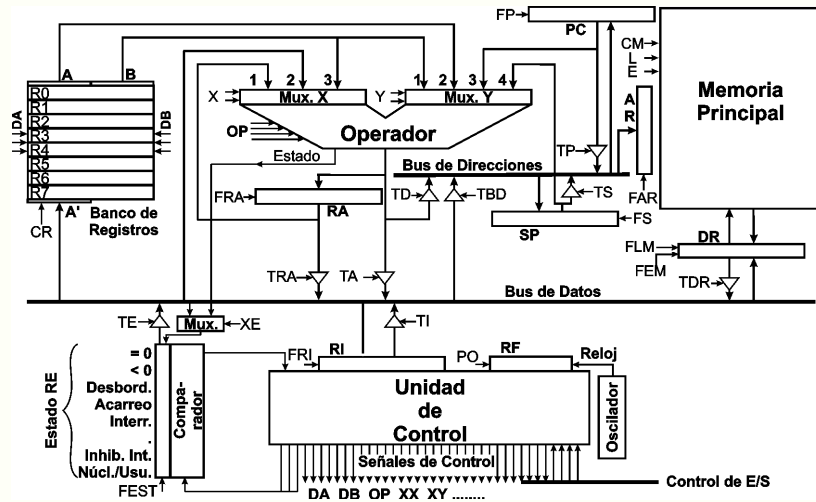
### ■ Índice

- Introducción
- Operaciones elementales
- Estructura de un computador elemental y sus señales de control
- Cronogramas
- Diseño de la UC
- UC microprogramada
- Control de excepciones

Unidad de Control

26

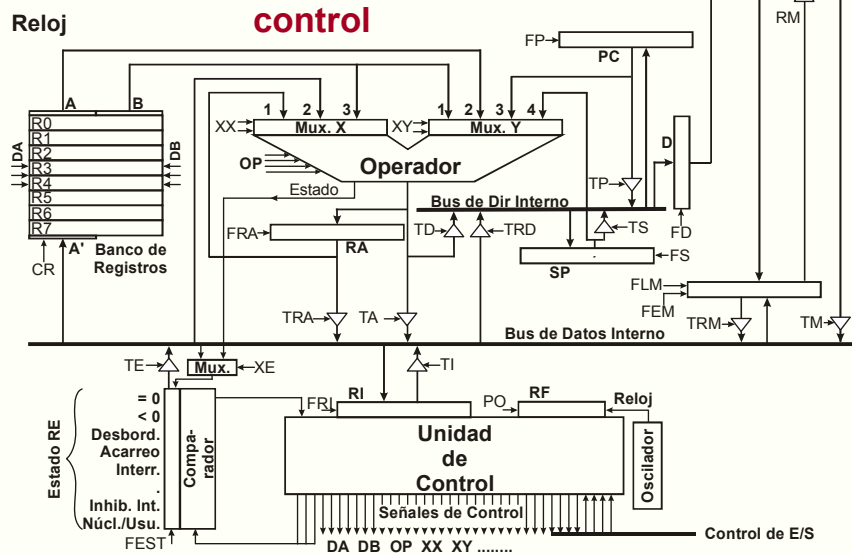
## Estructura de Computador Elemental y sus señales de control



Unidad de Control

27

## Estructura de Computador Elemental y sus señales de control



Unidad de Control

28



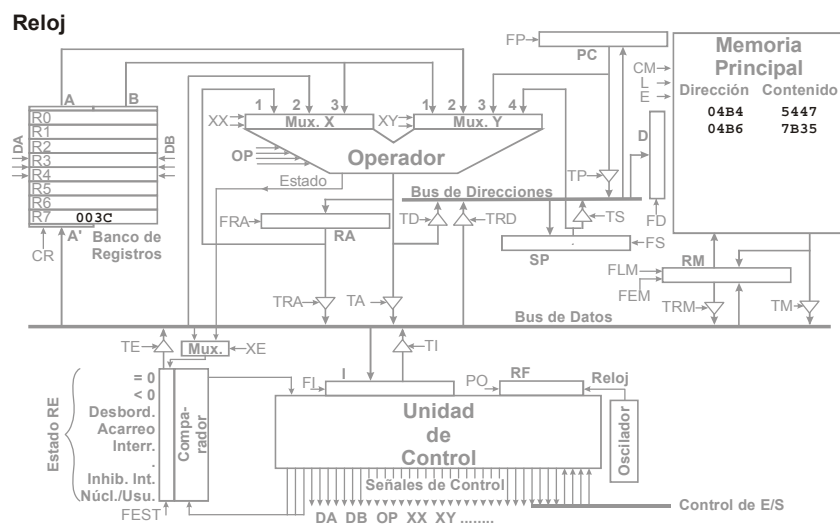
## Unidad de Control

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la UC
  - UC microprogramada
  - Control de excepciones

Unidad de Control

29

**ADD R4,R7**



Esta animación presenta la forma en que un computador sencillo ejecuta la instrucción de suma de dos registros.

Se supondrá que la arquitectura es de 16 bits y que la memoria se direcciona a nivel de byte. Además, el acceso a la memoria principal requiere dos ciclos.

La instrucción se encuentra almacenada en la posición de memoria H'4B4, ocupa 16 bits y se compone del código de operación (H'54) seguido de los descriptores de los registros, (4 y 7).

El resultado se dejará en el registro 4.

Los contenidos de los registros son H'4F2 y H'3C, por lo que el resultado es H'52E. Además, el único bit de estado que se activa es de NZ, puesto que el resultado no es cero.

La animación contempla dos imágenes por ciclo de reloj. La primera corresponde a las señales de control de tipo nivel, mientras que la segunda refleja las señales de carga.

Para ver el detalle de un ciclo se recomienda la animación "element", que muestra el desarrollo de una operación elemental.

Solamente se han indicado los valores de los registros afectados por la instrucción.

Se ha considerado que la lectura de la instrucción empieza en el ciclo 3 de la instrucción anterior y que el contador de ciclos se pone a "0" en el momento de la decodificación de la instrucción leída.

Las operaciones elementales necesarias son las siguientes:

$D \leftarrow PC$   
 $I \leftarrow M(D)$ ; esta lectura tarda dos ciclos  
Se decodifica la instrucción  
 $R4 \leftarrow R4 + R7$

Además de estas operaciones, se incrementa el contador de programa, para ir preparando la lectura de la siguiente instrucción.

$PC \leftarrow PC + 2$

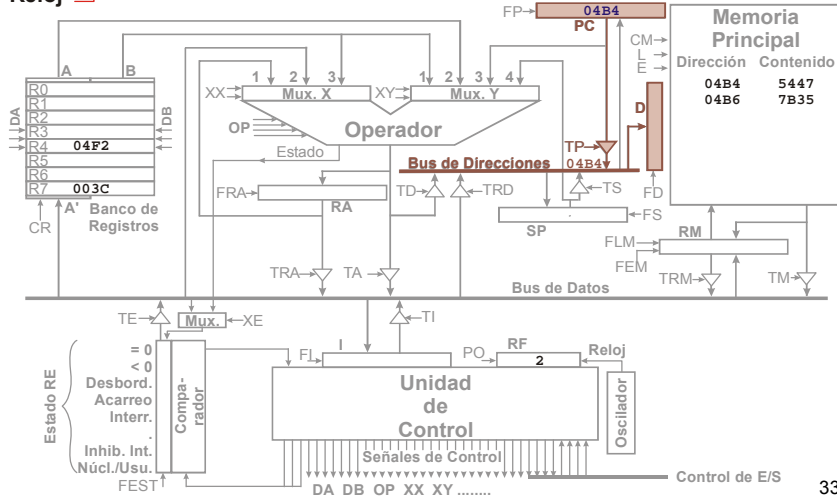


Prepara la lectura instrucción

Se prepara:  $D \leftarrow PC$

**ADD .R4,.R7**

Reloj 



33

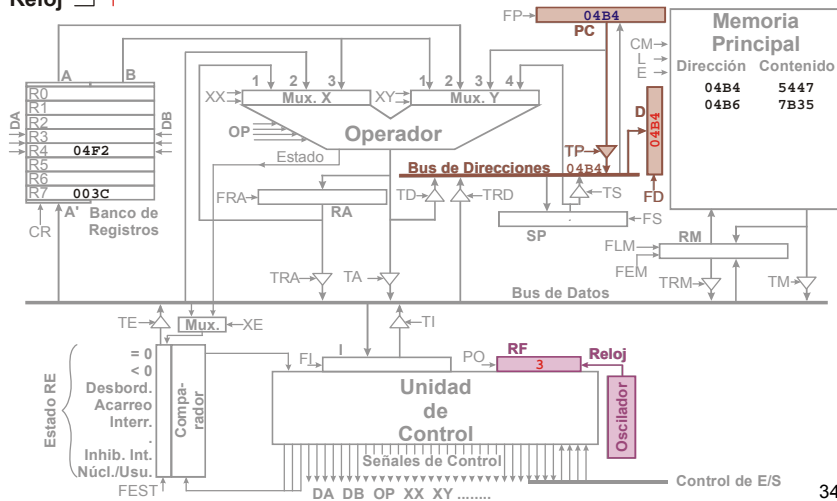
Prepara la lectura instrucción

Se realiza:  $D \leftarrow PC$

Se incrementa RF:  $RF \leftarrow RF + 1$

**ADD .R4,.R7**

Reloj 

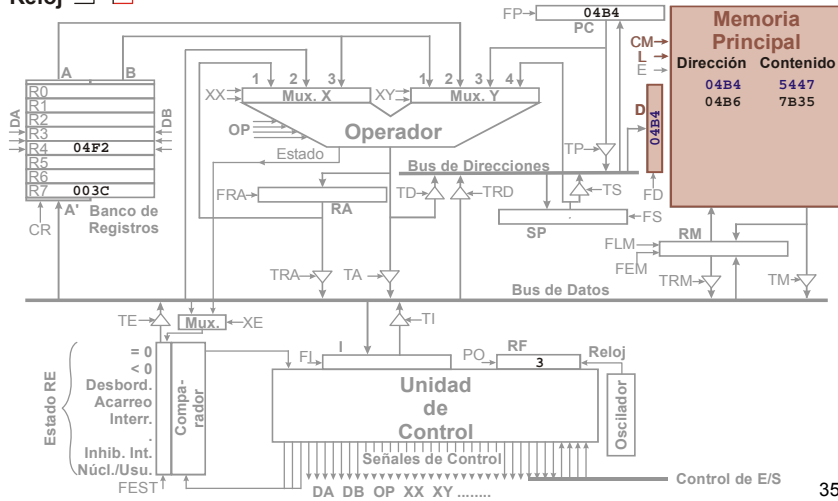


34

**Lectura instrucción**  
**Primer ciclo de memoria**

**ADD R4,R7**

Reloj 

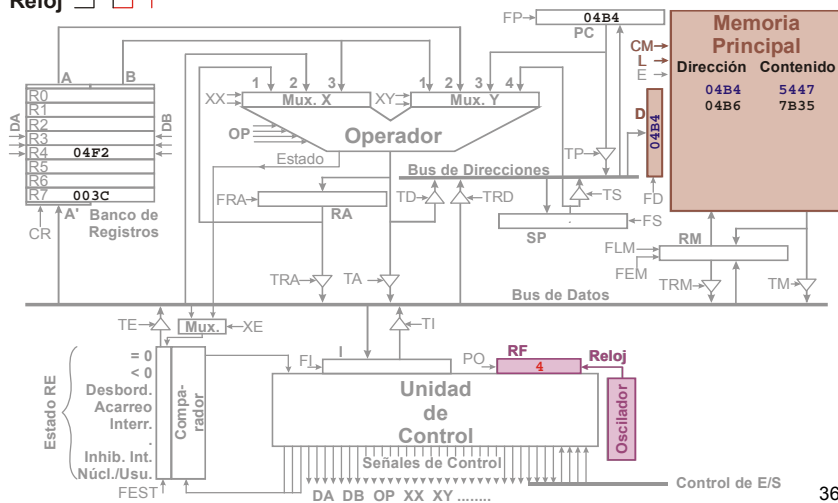


35

**Lectura instrucción**  
**Primer ciclo de memoria**  
**Se incrementa RF:  $RF \leftarrow RF + 1$**

**ADD R4,R7**

Reloj 



36

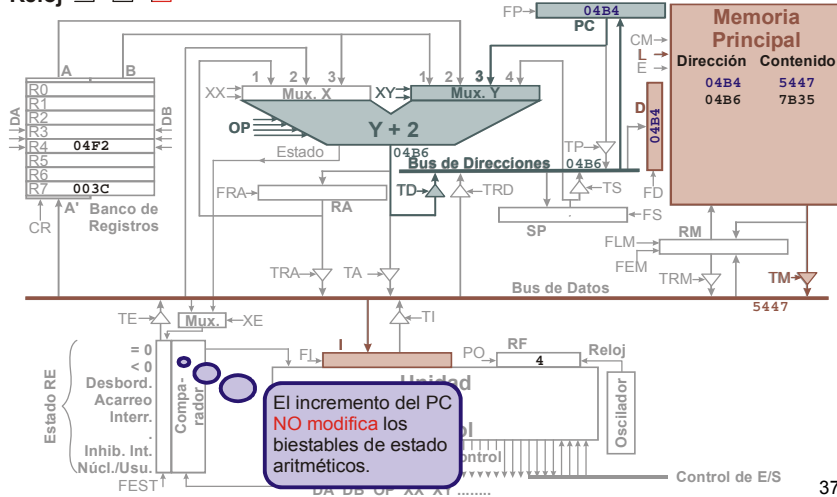
## ADD R4,R7

### Lectura instrucción

Segundo ciclo de memoria, se prepara:  $I \leftarrow M(D)$

Se prepara incremento de PC:  $PC \leftarrow PC + 2$

Reloj



37

## ADD R4,R7

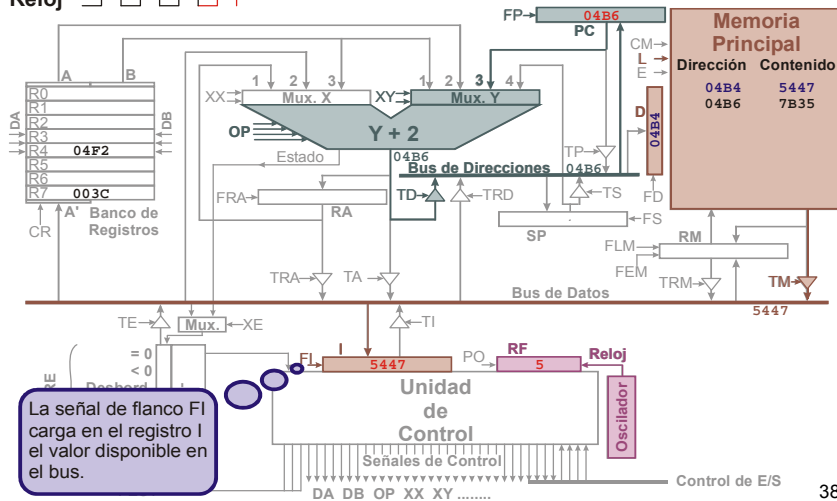
### Lectura instrucción

Segundo ciclo de memoria, se realiza:  $I \leftarrow M(D)$

Se realiza incremento de PC:  $PC \leftarrow PC + 2$

Se incrementa RF:  $RF \leftarrow RF + 1$

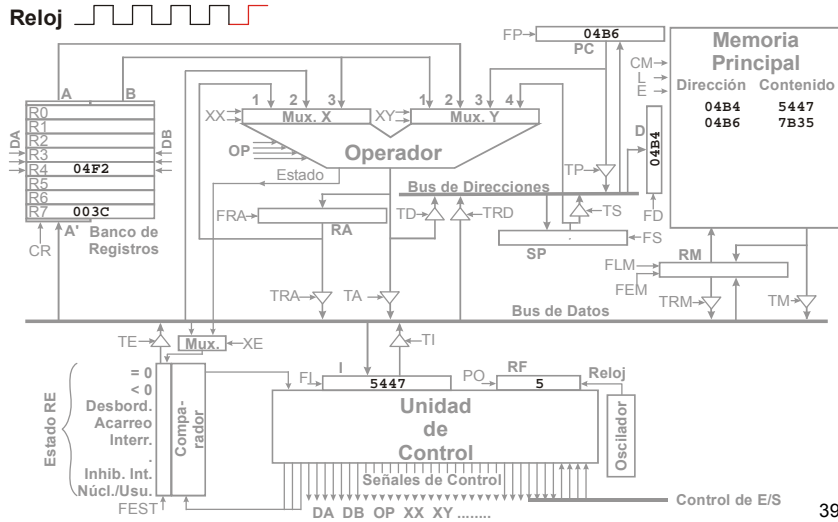
Reloj



38

## Decodificación instrucción

**ADD .R4,.R7**

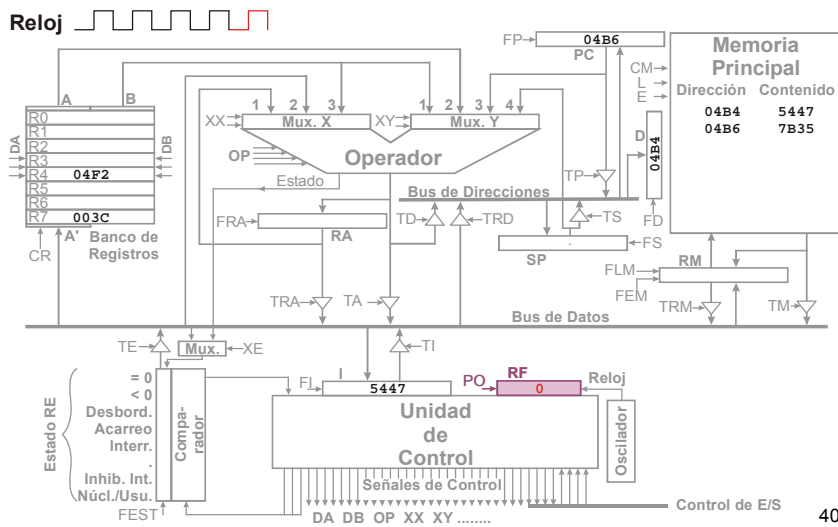


39

## Decodificación instrucción

Se pone RF a 0:  $RF \leftarrow 0$

**ADD .R4,.R7**



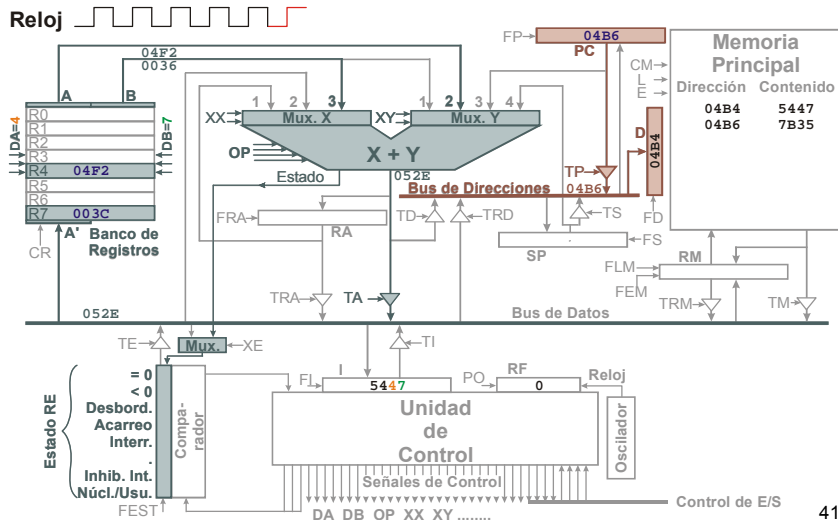
40

Ejecuta instrucción y prepara lectura instrucción siguiente

Se prepara suma:  $R4 \leftarrow R4 + R7$ ;  $RE \leftarrow$  Estado ALU

Se prepara:  $D \leftarrow PC$

**ADD .R4,.R7**



41

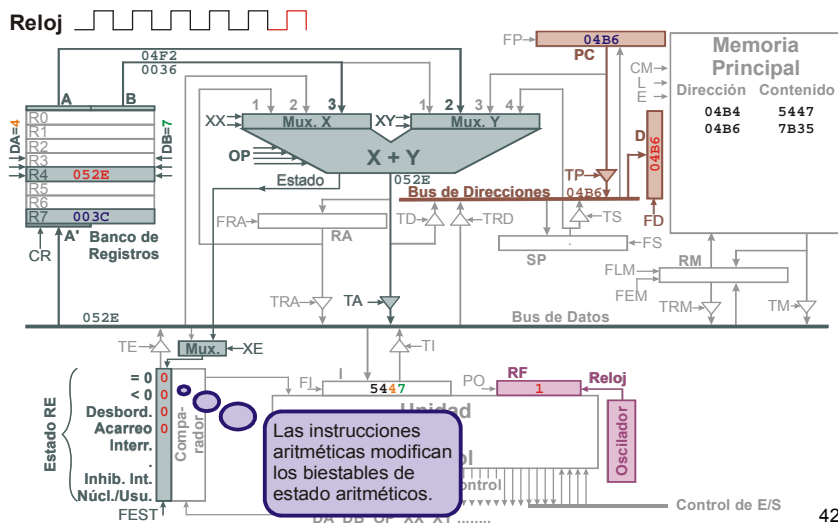
Ejecuta instrucción y prepara lectura instrucción siguiente

Se realiza suma:  $R4 \leftarrow R4 + R7$ ;  $RE \leftarrow$  Estado ALU

Se realiza:  $D \leftarrow PC$

Se incrementa RF:  $RF \leftarrow RF + 1$

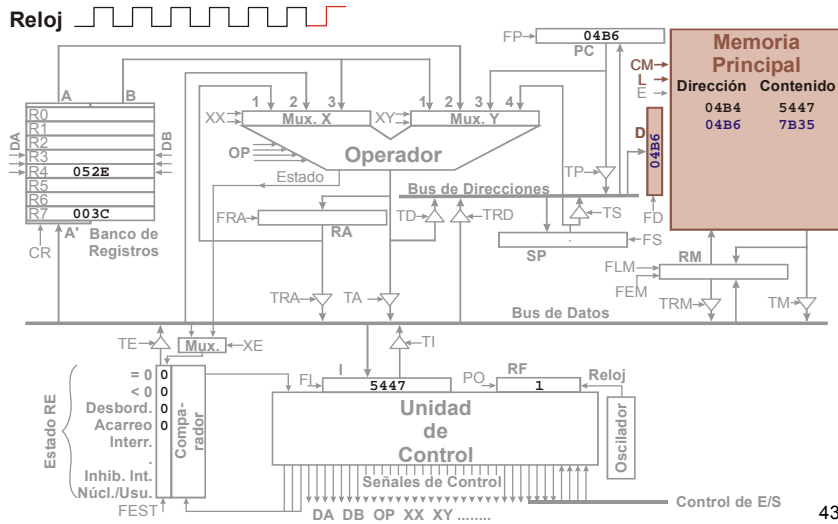
**ADD .R4,.R7**



42

Lectura instrucción siguiente  
Primer ciclo de memoria

**ADD R4,R7**

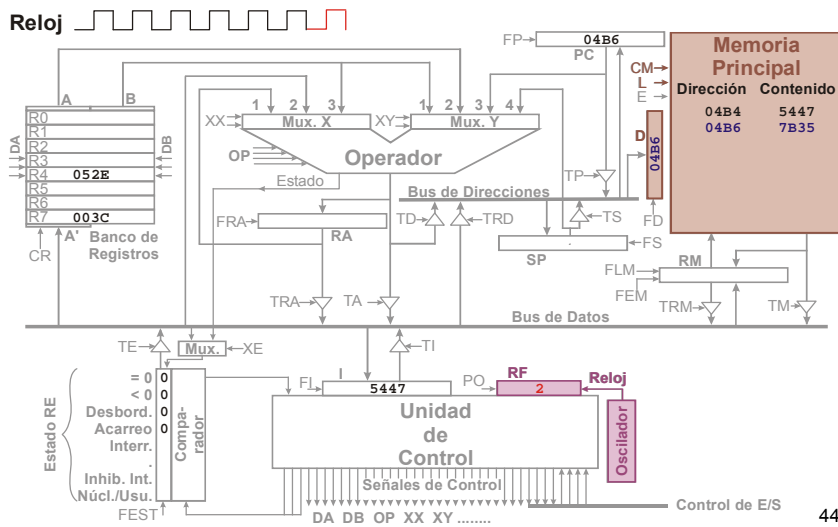


43

Lectura instrucción siguiente  
Primer ciclo de memoria

Se incrementa RF:  $RF \leftarrow RF + 1$

**ADD R4,R7**



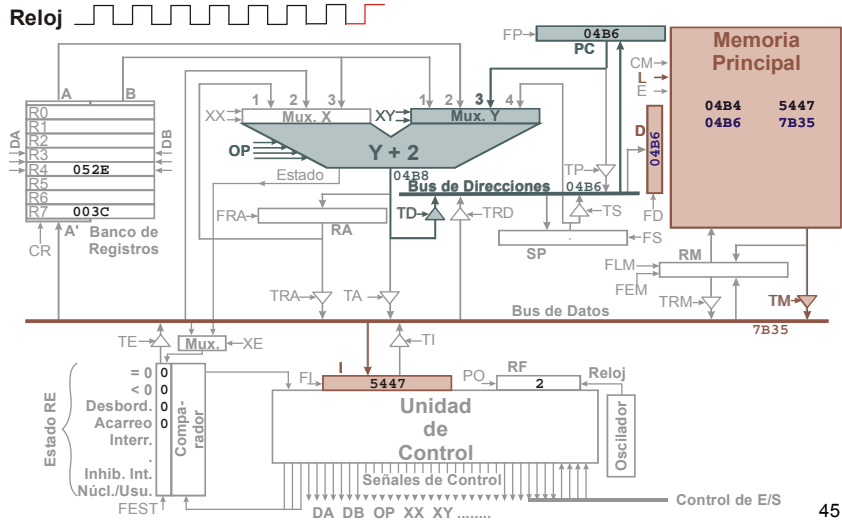
44

### Lectura instrucción

Segundo ciclo de memoria, se prepara:  $I \leftarrow M(D)$

Se prepara incremento de PC:  $PC \leftarrow PC + 2$

**ADD .R4.,R7**



45

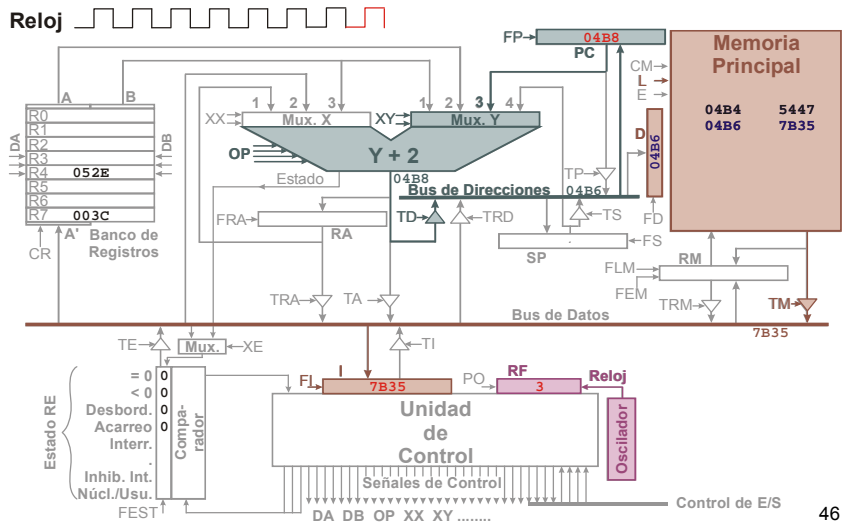
### Lectura instrucción

Segundo ciclo de memoria, se realiza:  $I \leftarrow M(D)$

Se realiza incremento de PC:  $PC \leftarrow PC + 2$

Se incrementa RF:  $RF \leftarrow RF + 1$

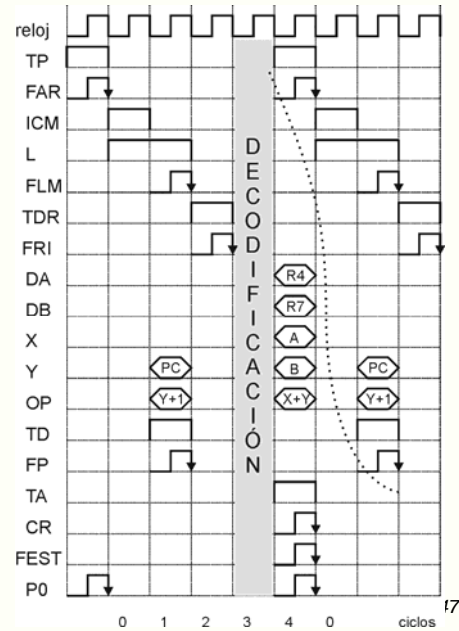
**ADD .R4.,R7**



46

## Cronogramas

- **ADD .R4, .R7**
- *Lectura de la instrucción*
  - PC → AR (1 ciclo)
  - M(AR) → DR, PC + 1 → PC (2 ciclos)
  - DR → RI (1 ciclo)
- *Decodificación (1 ciclo)*
- *Ejecución y fetch sig. Instrucción*
  - R4+R7 → R4 y FST;
  - PC → AR (1 ciclo)
  - M(AR) → DR, PC + 1 → PC (2 ciclos)
  - DR → RI (1 ciclo)



## Unidad de Control

- **Índice**
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - **Diseño de la UC**
  - UC microprogramada
  - Control de excepciones





## Diseño de la UC

- El diseño de la UC exige haber definido previamente
  - El repertorio de instrucciones (formatos y direccionamientos)
    - A nivel de cronogramas o de operaciones elementales
  - La estructura del computador
- La UC funciona como un TRADUCTOR  
RI(CO, MD), Reloj, Estado, señ ext → señales de control  
Ej: CO: 8 bits, 1 instr=16 ciclos → RF: 5 bits, ...  
traductor de unas 20-30 señales de entrada y 150 señales salida

Unidad de Control

49



## Diseño de la UC

- Formas de diseño de la UC
  - UC CABLEADA
    - Utilizando exclusivamente puertas lógicas
  - UC MICROPROGRAMADA
    - Utilizando una memoria de control

Unidad de Control

50



## Diseño de la UC

### ■ UC CABLEADA

- Se construye mediante métodos generales del diseño lógico (circuito secuencial o combinacional)
- Más rápidos
- Diseño y construcción complejo y costoso
- Las técnicas actuales (CAD, comp silicio) permiten que sea solución viable y válida para construir máquinas rápidas

Unidad de Control

51



## Diseño de la UC

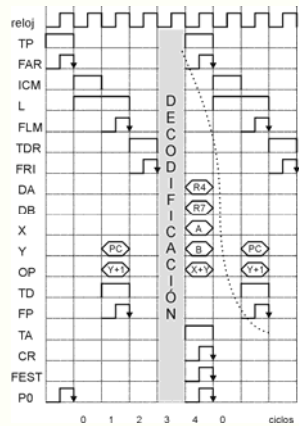
### ■ UC MICROPROGRAMADA

- Utiliza una memoria para almacenar el estado de las señales de control en cada periodo de la ejecución de cada instrucción.
- Generar el cronograma de cada instrucción (ejecutar la Inst) es ir leyendo de esta memoria (Memoria de Control).

Unidad de Control

52

## Diseño de la UC



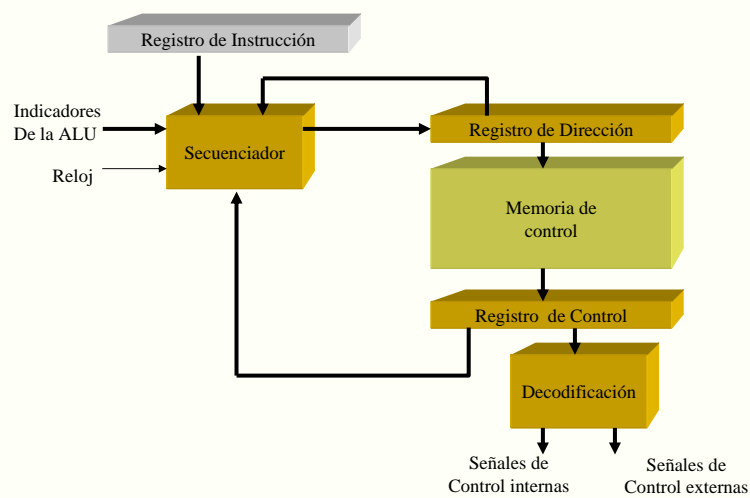
## Memoria de Control

TP	FA	IDA	L	FLM	TDR	DA	DB	FRI	X	Y	QF	TD	PP	TA	OR	FES	PO	...
1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1
1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
1	0	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
1	0	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0
x	x	x	x	x	x	0	1	1	0	0	0	1	0	0	0	1	1	0
x	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	0	0	0
x	x	x	x	x	x	0	1	0	1	0	0	0	0	0	0	1	1	0
x	x	x	x	x	x	0	1	1	0	0	0	0	0	0	0	0	1	0
x	x	x	x	x	x	0	1	1	0	0	0	0	1	1	1	0	0	0
x	x	x	x	x	x	0	1	1	0	0	0	0	0	0	0	0	0	0
x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	1	0	0	0
x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0
x	x	x	x	x	x	1	0	0	1	0	0	0	0	1	0	0	0	0

*Unidad de Control*

53

## Diseño de la UC



Unidad de Control

54

## Diseño de la UC

UC CABLEADA	UC MICROPROGRAMADA
<b>Ventajas</b> <ul style="list-style-type: none"> <li>Alta velocidad de funcionamiento</li> <li>Implementaciones más pequeñas (reducido número de componentes)</li> </ul>	<b>Ventajas</b> <ul style="list-style-type: none"> <li>Son sistemáticas con un formato bien definido</li> <li>Pueden ser fácilmente modificables durante el proceso de diseño</li> </ul>
<b>Inconvenientes</b> <ul style="list-style-type: none"> <li>Difícil realizar modificaciones en el diseño: modificación → rediseño</li> <li>No tienen estructura común</li> </ul>	<b>Inconvenientes</b> <ul style="list-style-type: none"> <li>Requieren más componentes para su implementación</li> <li>Tienden a ser más lentas que las unidades de control cableadas debido a que tienen que realizar operaciones de lectura de una memoria para obtener las señales de control</li> </ul>
<ul style="list-style-type: none"> <li>Técnica de diseño favorita en las arquitecturas <b>RISC</b></li> </ul>	<ul style="list-style-type: none"> <li><b>Emulación</b> <ul style="list-style-type: none"> <li>Se puede utilizar un microprograma para que la UC interprete otro lenguaje máquina distinto (una máquina a emular) sin necesidad de realizar modificaciones en el hardware de la unidad de control → cambiando sólo el microprograma!</li> </ul> </li> </ul>

## Diseño de la UC

- Conceptos básicos
  - UC MICROPROGRAMADA
  - MICROINSTRUCCIÓN
  - CICLO DE MÁQUINA
  - MICROPROGRAMA

## Conceptos básicos

### ■ Microinstrucción

- **Cadena de ceros y unos** que
  - representa la activación o no del conjunto de señales de control durante un ciclo de reloj
  - y la posibilidad de decidir condicionalmente qué microinstrucción se debe ejecutar a continuación
- **Cada palabra** de la memoria de control es una **microinstrucción**.

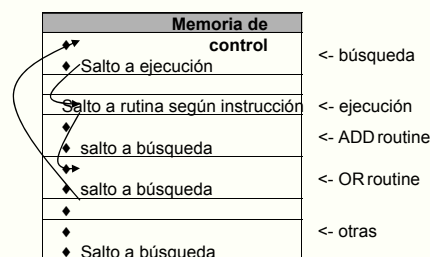
Unidad de Control

57

## Conceptos básicos

### ■ Microprograma

- Secuencia de  $\mu I$ 's cuya ejecución permite interpretar una instrucción
- Ejecutar una microinstrucción  $\approx$  Leer una  $\mu I$  de la memoria
- Una vez diseñado el camino de datos y la memoria de  $\mu I$ 's: el control es tarea de (micro)programación
- Cambiando microprogramas se puede ejecutar otros repertorios de instrucc (simular otras archit) -> **EMULACIÓN!**



Unidad de Control

58



## Conceptos básicos

- **Memoria de Control**
  - Memoria que almacena los microprogramas o (conjuntos de  $\mu I's$ ) a partir de los que se obtienen las señales de control necesarias para la ejecución del repertorio de instrucciones
  - Suelen ser ROM



## Diseño de la UC

### UC MICROPROGRAMADA

- Condiciones básicas:
  1. Capacidad suficiente de MC
  2. Procedimiento de correspondencia  
Inst  $\rightarrow$  dir MC comienzo su  $\mu$ programa
  3. Secuenciamiento de  $\mu$ instrucciones



## Conceptos básicos

### ■ Formato de microinstrucciones

- **Señales de control:** Señales para el camino de datos
- **Condiciones:** Bits para seleccionar la condición que se desea utilizar para en función de si es cierta o no ejecutar la siguiente microinstrucción o saltar a otra
- **Siguiente  $\mu$ instrucción:** Campo que indica la siguiente  $\mu$ instrucción a ejecutar

Señales de control	Condiciones	Siguiente $\mu$ instrucción
--------------------	-------------	-----------------------------

Unidad de Control

61



## Diseño de la UC

### Diseño del formato de microinstrucción

- Comenzar con la lista completa de señales de control
- Agrupar las señales de control con función similar en un mismo **CAMPO**
  - Controlar que una misma señal no esté en diferentes campos
- Campo de **CONTROL DE SECUENCIAMIENTO:** indica la siguiente  $\mu$ l a ejecutar
- Secuenciamiento: se deben almacenar en algún orden en MC, por ejemplo, secuencialmente (sec. implícito)
- Se puede aplicar CODIFICACIÓN:
  - Para reducir el tamaño de la  $\mu$ l
  - Para evitar que ciertas señales se activen al mismo tiempo (evitar errores)

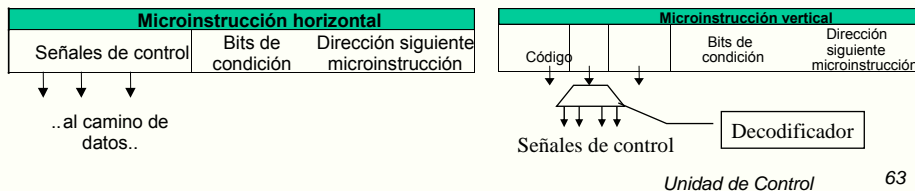
Ej: acceso a bus, operación de memoria, etc...

Unidad de Control

62

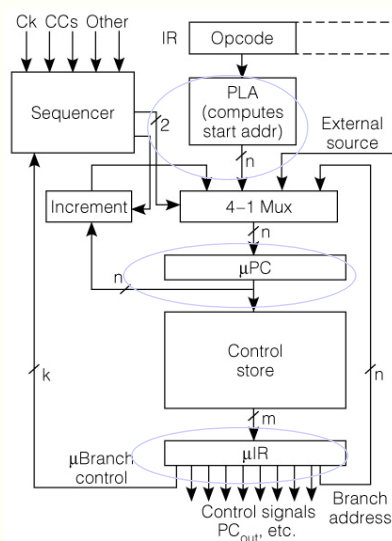
## Conceptos básicos

- **Codificación de los campos de la microinstrucciones**
  - **Microinstrucción horizontal:** Campos no codificados. Las señales de control se obtienen directamente de los bits de la microinstrucción.
    - Pocas microinstrucciones y anchas. Permite realizar acciones en paralelo
  - **Microinstrucción vertical:** Campos codificados. Para obtener las señales de control necesitamos un decodificador.
    - Muchas microinstrucciones y cortas. No permite acciones en paralelo.
  - **Diseños intermedios**



## Diseño de la UC

- **Estructura básica de la UC microprogramada**



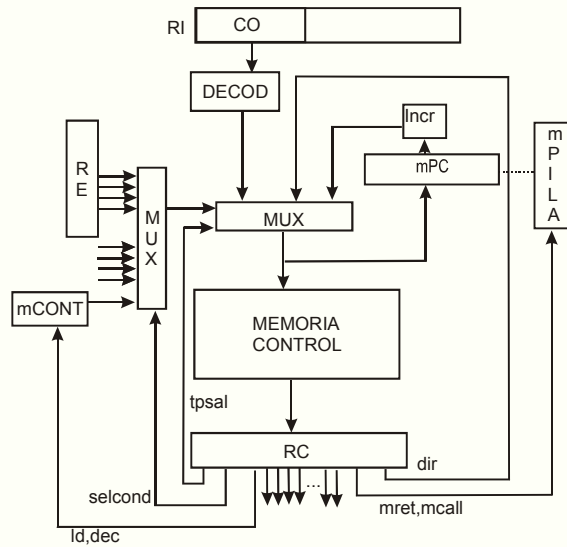
Copyright © 2004 Pearson Prentice Hall, Inc.



## Diseño de la UC

Estructura completa  
de la UC  
mprogramada

- microbucles
- microsubsutinas



Unidad de Control

65

## Unidad de Control

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la UC
  - UC microprogramada
  - Control de excepciones

Unidad de Control

66



## Control de excepciones

- Además de ejecutar la secuencia de instrucciones del programa de usuario, la UC debe controlar situaciones excepcionales
  - Qué ocurre si el CO no es válido?
  - Y si hay overflow?
  - Cómo y cuándo comunicar con algún periférico?

Unidad de Control

67



## Control de excepciones

- Excepción [Overflow...]
  - Evento no planificado que para la ejecución de un programa. (interno)
    - Salvar el PC y Estado actual en pila
    - Saltar a rutina de SO que maneje excepciones internas
    - SO retornará después al programa, en su anterior estado
- Interrupción [E/S]
  - Un evento externo inesperado rompe la ejecución del programa

Unidad de Control

68



## Control de excepciones

- Objetivos
  - *Detección de excepciones* – Cuándo y cómo verlas?
  - *Manejo de excepciones* – Qué hacer?



## Control de excepciones

### Tipos de excepciones

- Interrupciones:
  - Causadas por eventos externos (peticiones de periféricos).
  - Asíncronos a la ejecución del programa (cualquier instante)
  - Se atienden entre Instrucciones (fetch)
  - Mantienen sin ejecutarse un tiempo el programa de usuario



## Control de excepciones

### Tipos de excepciones

#### ■ Traps:

- Causadas por eventos internos
  - Condiciones de excepción (ej: overflow)
  - Errores (ej: error de paridad)
  - Fallos (ej: Fallo de pagina)
  - Llamadas al sistema
- Síncronos a la ejecución del programa (se puede saber cuándo)
- Puede que el programa de usuario se retome tras tratar la excepción, o bien que se aborte

Unidad de Control

71



## Control de excepciones



### Manejo de excepciones

- #### ■ Excepción = transferencia de control no programada

El sistema maneja la excepción:

- Guarda la dir de la Instrucción que causó la excepción
- Salvaguarda el estado del programa de usuario
- Retorna el control a usuario (salvo que se aborte)

Unidad de Control

72



## Control de excepciones

### ◆ Interrupciones Vectorizadas:

1. Cada excepción tiene una dirección diferente asociada
  2. Detectada la excepción => la CPU calcula para esa **excepción** la dirección a escribir en PC (depende del periférico, que pone un VECTOR en bus).
- Cuando atenderlas?
    - Entre Instrucciones: antes de comenzar el FETCH

*FETCH\*\**: Si INT e INT\_no\_inhibidas ir a  
Microrrut\_proceso\_excepcion  
Si no ir a FETCH